Mini Project 2

Problem Statement: Which model is suitable for FlightPricePredection dataset

Importing Packages

In [172]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

Reading datasets

In [173]:
```
traindf=pd.read_csv(r"C:\Users\P. VIJAY KUMAR\Downloads\Copy of Data_Trai
traindf
```

Out[173]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration
0	IndiGo	24/03/2019	Banglore	New Delhi	BLR → DEL	22:20	01:10 22 Mar	2h 50m
1	Air India	1/05/2019	Kolkata	Banglore	CCU → IXR → BBI → BLR	05:50	13:15	7h 25m
2	Jet Airways	9/06/2019	Delhi	Cochin	DEL → LKO → BOM → COK	09:25	04:25 10 Jun	19h
3	IndiGo	12/05/2019	Kolkata	Banglore	CCU → NAG → BLR	18:05	23:30	5h 25m
4	IndiGo	01/03/2019	Banglore	New Delhi	BLR → NAG → DEL	16:50	21:35	4h 45m
...	...	...	...	...	...	...	...	..
10678	Air Asia	9/04/2019	Kolkata	Banglore	CCU → BLR	19:55	22:25	2h 30m
10679	Air India	27/04/2019	Kolkata	Banglore	CCU → BLR	20:45	23:20	2h 35m
10680	Jet Airways	27/04/2019	Banglore	Delhi	BLR → DEL	08:20	11:20	3h
10681	Vistara	01/03/2019	Banglore	New Delhi	BLR → DEL	11:30	14:10	2h 40m
10682	Air India	9/05/2019	Delhi	Cochin	DEL → GOI → BOM → COK	10:55	19:15	8h 20m

10683 rows × 11 columns

In [174]:
```
testdf=pd.read_csv(r"C:\Users\P. VIJAY KUMAR\Downloads\Copy of Test_set.cs
testdf
```

Out[174]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration
0	Jet Airways	6/06/2019	Delhi	Cochin	DEL → BOM → COK	17:30	04:25 07 Jun	10h 55m
1	IndiGo	12/05/2019	Kolkata	Banglore	CCU → MAA → BLR	06:20	10:20	4h
2	Jet Airways	21/05/2019	Delhi	Cochin	DEL → BOM → COK	19:15	19:00 22 May	23h 45m
3	Multiple carriers	21/05/2019	Delhi	Cochin	DEL → BOM → COK	08:00	21:00	13h
4	Air Asia	24/06/2019	Banglore	Delhi	BLR → DEL	23:55	02:45 25 Jun	2h 50m
...	...	...	...	...	...	...	...	...
2666	Air India	6/06/2019	Kolkata	Banglore	CCU → DEL → BLR	20:30	20:25 07 Jun	23h 55m
2667	IndiGo	27/03/2019	Kolkata	Banglore	CCU → BLR	14:20	16:55	2h 35m
2668	Jet Airways	6/03/2019	Delhi	Cochin	DEL → BOM → COK	21:50	04:25 07 Mar	6h 35m
2669	Air India	6/03/2019	Delhi	Cochin	DEL → BOM → COK	04:00	19:15	15h 15m
2670	Multiple carriers	15/06/2019	Delhi	Cochin	DEL → BOM → COK	04:55	19:15	14h 20m

2671 rows × 10 columns

Data Preprocessing

In [77]:
```
1 traindf.isnull().sum()
```

Out[77]:
```
Airline           0
Date_of_Journey   0
Source            0
Destination       0
Route             1
Dep_Time          0
Arrival_Time      0
Duration          0
Total_Stops       1
Additional_Info   0
Price             0
dtype: int64
```

In [ ]:
```
1
```

In [78]:
```
1 traindf.dropna(inplace=True)
```

In [79]:
```
1 traindf.isnull().sum()
```

Out[79]:
```
Airline           0
Date_of_Journey   0
Source            0
Destination       0
Route             0
Dep_Time          0
Arrival_Time      0
Duration          0
Total_Stops       0
Additional_Info   0
Price             0
dtype: int64
```

In [81]:
```
1 traindf.head()
```

Out[81]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration	Tota
0	2	24/03/2019	3	4	BLR → DEL	22:20	01:10 22 Mar	2h 50m	
1	3	1/05/2019	2	2	CCU → IXR → BBI → BLR	05:50	13:15	7h 25m	
2	1	9/06/2019	1	1	DEL → LKO → BOM → COK	09:25	04:25 10 Jun	19h	
3	2	12/05/2019	2	2	CCU → NAG → BLR	18:05	23:30	5h 25m	
4	2	01/03/2019	3	4	BLR → NAG → DEL	16:50	21:35	4h 45m	

In [82]:
```
1 testdf.head()
```

Out[82]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration	To
0	Jet Airways	6/06/2019	Delhi	Cochin	DEL → BOM → COK	17:30	04:25 07 Jun	10h 55m	
1	IndiGo	12/05/2019	Kolkata	Banglore	CCU → MAA → BLR	06:20	10:20	4h	
2	Jet Airways	21/05/2019	Delhi	Cochin	DEL → BOM → COK	19:15	19:00 22 May	23h 45m	
3	Multiple carriers	21/05/2019	Delhi	Cochin	DEL → BOM → COK	08:00	21:00	13h	
4	Air Asia	24/06/2019	Banglore	Delhi	BLR → DEL	23:55	02:45 25 Jun	2h 50m	

In [83]:
```
1 traindf.shape,testdf.shape
```

Out[83]:
```
((10682, 11), (2671, 10))
```

In [84]:
```
1 traindf.columns
```

Out[84]:
```
Index(['Airline', 'Date_of_Journey', 'Source', 'Destination', 'Route',
       'Dep_Time', 'Arrival_Time', 'Duration', 'Total_Stops',
       'Additional_Info', 'Price'],
      dtype='object')
```

In [85]:
```
1 traindf["Airline"].value_counts()
```

Out[85]:
```
Airline
1     3849
2     2053
3     1751
4     1196
5      818
6      479
7      319
8      194
9       13
10       6
11       3
12       1
Name: count, dtype: int64
```

In [86]:

```
convert={"Airline":{"Jet Airways":1,"IndiGo":2,"Air India":3,"Multiple ca
                    "Vistara":6,"Air Asia":7,"GoAir":8,"Multiple carriers
                    "Jet Airways Business":10,"Vistara Premium economy":11
traindf=traindf.replace(convert)
traindf
```

Out[86]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration
0	2	24/03/2019	3	4	BLR → DEL	22:20	01:10 22 Mar	2h 50m
1	3	1/05/2019	2	2	CCU → IXR → BBI → BLR	05:50	13:15	7h 25m
2	1	9/06/2019	1	1	DEL → LKO → BOM → COK	09:25	04:25 10 Jun	19h
3	2	12/05/2019	2	2	CCU → NAG → BLR	18:05	23:30	5h 25m
4	2	01/03/2019	3	4	BLR → NAG → DEL	16:50	21:35	4h 45m
...	...	...	...	...	...	...	...	...
10678	7	9/04/2019	2	2	CCU → BLR	19:55	22:25	2h 30m
10679	3	27/04/2019	2	2	CCU → BLR	20:45	23:20	2h 35m
10680	1	27/04/2019	3	3	BLR → DEL	08:20	11:20	3h
10681	6	01/03/2019	3	4	BLR → DEL	11:30	14:10	2h 40m
10682	3	9/05/2019	1	1	DEL → GOI → BOM → COK	10:55	19:15	8h 20m

10682 rows × 11 columns

In [87]:
```
1 traindf["Source"].value_counts()
```

Out[87]:
```
Source
1    4536
2    2871
3    2197
4     697
5     381
Name: count, dtype: int64
```

In [88]:
```
convert={"Source":{"Delhi":1,"Kolkata":2,"Banglore":3,"Mumbai":4,"Chennai"
traindf=traindf.replace(convert)
traindf
```

Out[88]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration
0	2	24/03/2019	3	4	BLR → DEL	22:20	01:10 22 Mar	2h 50m
1	3	1/05/2019	2	2	CCU → IXR → BBI → BLR	05:50	13:15	7h 25m
2	1	9/06/2019	1	1	DEL → LKO → BOM → COK	09:25	04:25 10 Jun	19h
3	2	12/05/2019	2	2	CCU → NAG → BLR	18:05	23:30	5h 25m
4	2	01/03/2019	3	4	BLR → NAG → DEL	16:50	21:35	4h 45m
...	...	...	...	...	...	...	...	...
10678	7	9/04/2019	2	2	CCU → BLR	19:55	22:25	2h 30m
10679	3	27/04/2019	2	2	CCU → BLR	20:45	23:20	2h 35m
10680	1	27/04/2019	3	3	BLR → DEL	08:20	11:20	3h
10681	6	01/03/2019	3	4	BLR → DEL	11:30	14:10	2h 40m
10682	3	9/05/2019	1	1	DEL → GOI → BOM → COK	10:55	19:15	8h 20m

10682 rows × 11 columns

In [89]:
```
1 traindf["Destination"].value_counts()
```

Out[89]:
```
Destination
1    4536
2    2871
3    1265
4     932
5     697
6     381
Name: count, dtype: int64
```

In [90]:
```
convert={"Destination":{"Cochin":1,"Banglore":2,"Delhi":3,"New Delhi":4,"
traindf=traindf.replace(convert)
traindf
```

Out[90]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration
0	2	24/03/2019	3	4	BLR → DEL	22:20	01:10 22 Mar	2h 50m
1	3	1/05/2019	2	2	CCU → IXR → BBI → BLR	05:50	13:15	7h 25m
2	1	9/06/2019	1	1	DEL → LKO → BOM → COK	09:25	04:25 10 Jun	19h
3	2	12/05/2019	2	2	CCU → NAG → BLR	18:05	23:30	5h 25m
4	2	01/03/2019	3	4	BLR → NAG → DEL	16:50	21:35	4h 45m
...	...	...	...	...	...	...	...	...
10678	7	9/04/2019	2	2	CCU → BLR	19:55	22:25	2h 30m
10679	3	27/04/2019	2	2	CCU → BLR	20:45	23:20	2h 35m
10680	1	27/04/2019	3	3	BLR → DEL	08:20	11:20	3h
10681	6	01/03/2019	3	4	BLR → DEL	11:30	14:10	2h 40m
10682	3	9/05/2019	1	1	DEL → GOI → BOM → COK	10:55	19:15	8h 20m

10682 rows × 11 columns

In [91]:
```
1 traindf["Total_Stops"].value_counts()
```

Out[91]:
```
Total_Stops
1.0    5626
2.0    3491
3.0    1520
4.0      45
Name: count, dtype: int64
```

In [92]:
```
convert={"Total_Stops":{"1 stop":1,"non-stop":2,"2 stops":3,"3 stops":4,"
traindf=traindf.replace(convert)
traindf
```

Out[92]:

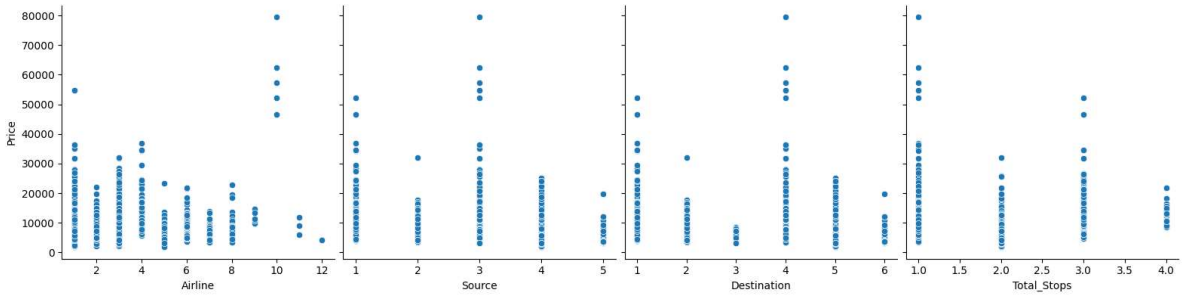
	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration
0	2	24/03/2019	3	4	BLR → DEL	22:20	01:10 22 Mar	2h 50m
1	3	1/05/2019	2	2	CCU → IXR → BBI → BLR	05:50	13:15	7h 25m
2	1	9/06/2019	1	1	DEL → LKO → BOM → COK	09:25	04:25 10 Jun	19h
3	2	12/05/2019	2	2	CCU → NAG → BLR	18:05	23:30	5h 25m
4	2	01/03/2019	3	4	BLR → NAG → DEL	16:50	21:35	4h 45m
...	...	...	...	...	...	...	...	...
10678	7	9/04/2019	2	2	CCU → BLR	19:55	22:25	2h 30m
10679	3	27/04/2019	2	2	CCU → BLR	20:45	23:20	2h 35m
10680	1	27/04/2019	3	3	BLR → DEL	08:20	11:20	3h
10681	6	01/03/2019	3	4	BLR → DEL	11:30	14:10	2h 40m
10682	3	9/05/2019	1	1	DEL → GOI → BOM → COK	10:55	19:15	8h 20m

10682 rows × 11 columns

Data Visualization

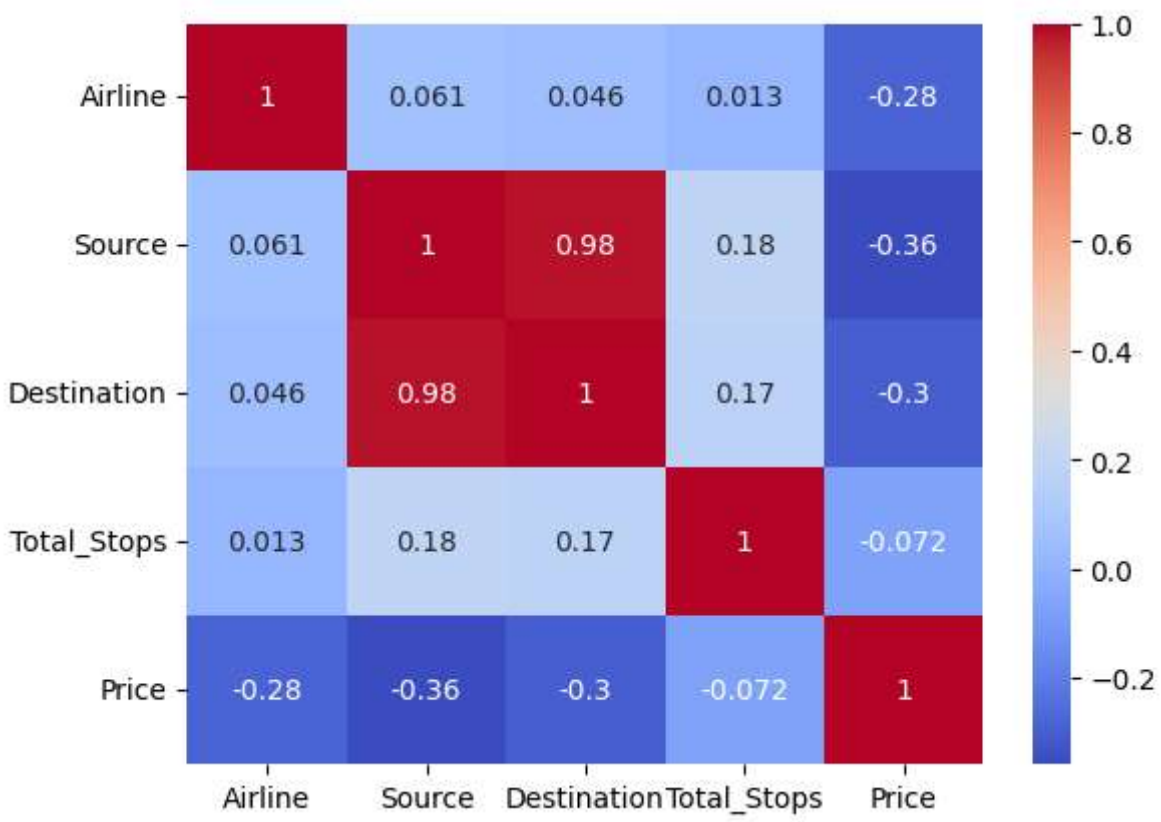
In [93]:
```
1 sns.pairplot(traindf,x_vars=["Airline","Source","Destination","Total_Stops
```

Out[93]: <seaborn.axisgrid.PairGrid at 0x262597a4880>



In [94]:
```
1 columns_to_plot=traindf[["Airline","Source","Destination","Total_Stops","
2 columns=columns_to_plot.corr()
3 sns.heatmap(columns,annot=True,cmap='coolwarm')
```

Out[94]: <Axes: >



Feature Scaling: Splitting dataset into training and testing datasets

In [116]:
```
x=np.array(traindf["Price"]).reshape(-1,1)
y=np.array(traindf["Total_Stops"]).reshape(-1,1)
```

In [117]:
```
from sklearn.model_selection import train_test_split
```

In [118]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3,random_s
```

Applying Linear Regression

In [119]:
```
from sklearn.linear_model import LinearRegression
```

In [120]:
```
lr=LinearRegression()
```

In [121]:
```
lr.fit(x_train,y_train)
```

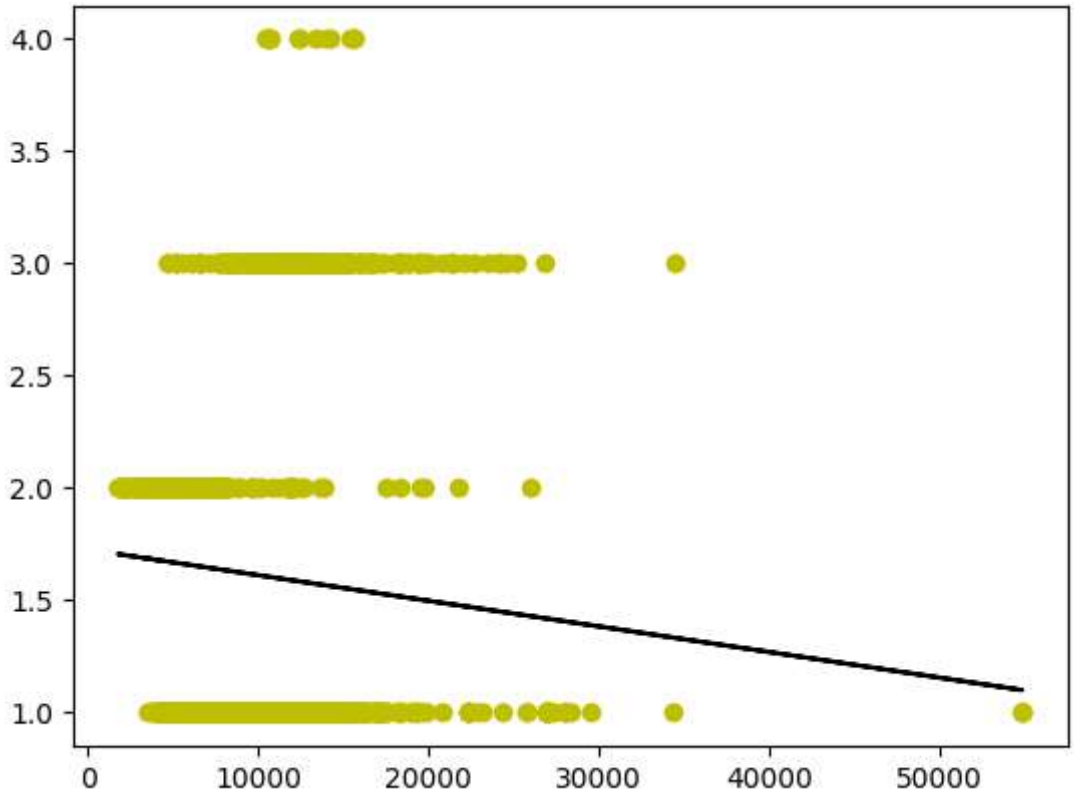
Out[121]:
```
LinearRegression()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [122]:
```
lr.score(x_test,y_test)
```

Out[122]:
```
0.0052708846920045405
```

In [123]:
```
prediction=lr.predict(x_test)
plt.scatter(x_test,y_test,color='y')
plt.plot(x_test,prediction,color='k')
plt.show()
```



Applying Logistic Regression

In [175]:
```
from sklearn.linear_model import LogisticRegression
```

In [124]:
```
#LogisticRegression
lg=LogisticRegression()
```

In [125]:
```
X=y
Y=x
```

In [126]:
```
1 lg.fit(x_train,y_train)
```

```
C:\Users\P. VIJAY KUMAR\AppData\Roaming\Python\Python310\site-packages\sklearn\utils\validation.py:1143: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

Out[126]:
```
LogisticRegression()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

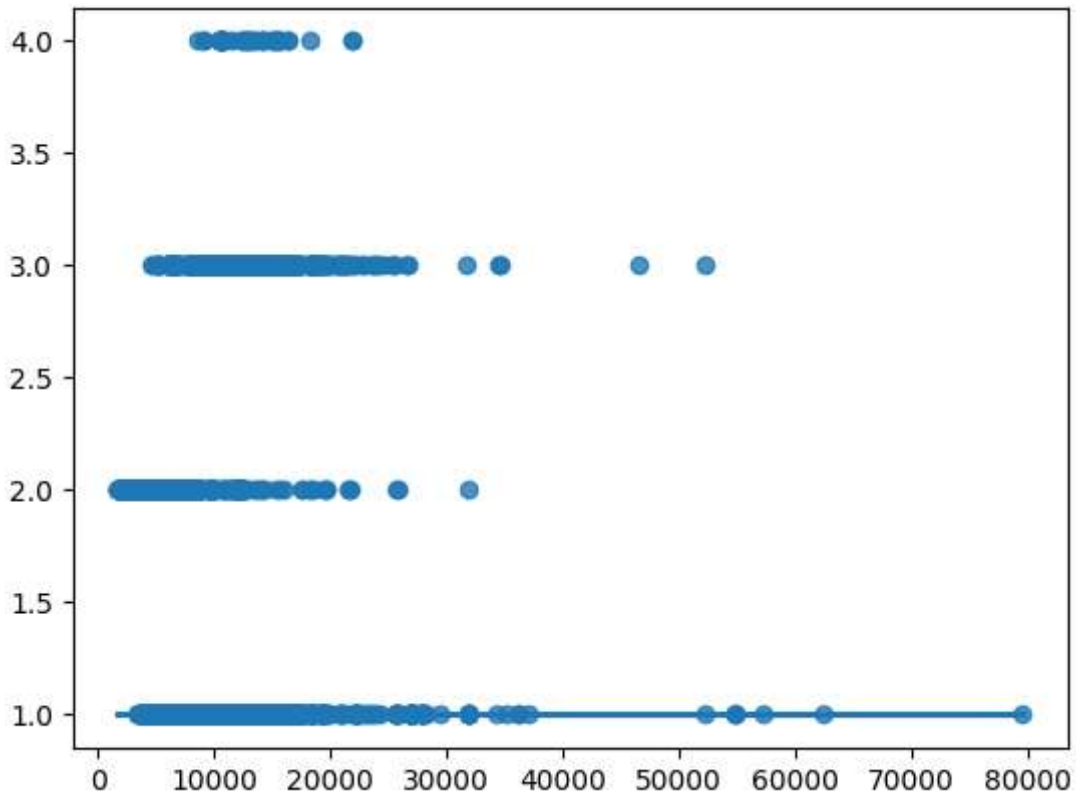
In [127]:
```
1 lg.score(x_test,y_test)
```

Out[127]:
```
0.7160686427457098
```

In [129]:
```
1 sns.regplot(x=x,y=y,data=traindf,logistic=True,ci=None)
```

```
C:\Users\P. VIJAY KUMAR\AppData\Roaming\Python\Python310\site-packages\statsmodels\genmod\families\links.py:198: RuntimeWarning: overflow encountered in exp
  t = np.exp(-z)
```

Out[129]:
```
<Axes: >
```



Applying Decision Tree

In [130]:
```
#Decision Tree
from sklearn.tree import DecisionTreeClassifier
dtl=DecisionTreeClassifier(random_state=0)
```

In [131]:
```
dtl.fit(x_train,y_train)
```

Out[131]:
```
DecisionTreeClassifier(random_state=0)
```
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [148]:
```
dtl.score(x_test,y_test)
```

Out[148]:
```
0.9341653666146645
```

Applying Random Forest

In [161]:
```
#Random Forest
from sklearn.ensemble import RandomForestClassifier
rfc=RandomForestClassifier()
```

In [162]:
```
rfc.fit(x_train,y_train)
```

```
C:\Users\P. VIJAY KUMAR\AppData\Local\Temp\ipykernel_4340\4070307935.py:1: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example using ravel().
  rfc.fit(x_train,y_train)
```

Out[162]:
```
RandomForestClassifier()
```
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [163]:
```
rfc.score(x_test,y_test)
```

Out[163]:
```
0.9341653666146645
```

In [164]:
```
params={'max_depth':[2,3,5,10,20],
        'min_samples_leaf':[5,10,20,50,100,200],
        'n_estimators':[10,25,30,50,100,200]}
```

In [165]:
```
from sklearn.model_selection import GridSearchCV
grid_search=GridSearchCV(estimator=rfc,param_grid=params,cv=2,scoring="ac
```

In [166]:
```
grid_search.fit(x_train,y_train)
```

```
C:\Users\P. VIJAY KUMAR\AppData\Roaming\Python\Python310\site-packages\skl
earn\model_selection\_validation.py:686: DataConversionWarning: A column-v
ector y was passed when a 1d array was expected. Please change the shape o
f y to (n_samples,), for example using ravel().
  estimator.fit(X_train, y_train, **fit_params)
C:\Users\P. VIJAY KUMAR\AppData\Roaming\Python\Python310\site-packages\skl
earn\model_selection\_validation.py:686: DataConversionWarning: A column-v
ector y was passed when a 1d array was expected. Please change the shape o
f y to (n_samples,), for example using ravel().
  estimator.fit(X_train, y_train, **fit_params)
C:\Users\P. VIJAY KUMAR\AppData\Roaming\Python\Python310\site-packages\skl
earn\model_selection\_validation.py:686: DataConversionWarning: A column-v
ector y was passed when a 1d array was expected. Please change the shape o
f y to (n_samples,), for example using ravel().
  estimator.fit(X_train, y_train, **fit_params)
C:\Users\P. VIJAY KUMAR\AppData\Roaming\Python\Python310\site-packages\skl
earn\model_selection\_validation.py:686: DataConversionWarning: A column-v
ector y was passed when a 1d array was expected. Please change the shape o
f y to (n_samples,), for example using ravel().
```

In [167]:
```
grid_search.best_score_
```

Out[167]:
```
0.8769559604195134
```

In [168]:
```
rf_best=grid_search.best_estimator_
rf_best
```

Out[168]:
```
RandomForestClassifier(max_depth=20, min_samples_leaf=5, n_estimators=50)
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [169]:
```
from sklearn.tree import plot_tree
plt.figure(figsize=(80,40))
plot_tree(rf_best.estimators_[4],class_names=['0','1','2','3','4'],filled
```

Out[169]:
```
[Text(0.2790515376072304, 0.975, 'x[0] <= 5786.5\ngini = 0.594\nsamples =
4689\nvalue = [3951, 2440, 1051, 35]\nclass = 0'),
 Text(0.09426062091503268, 0.925, 'x[0] <= 4950.5\ngini = 0.22\nsamples =
1364\nvalue = [271, 1891, 1, 0]\nclass = 1'),
 Text(0.049836601307189546, 0.875, 'x[0] <= 4200.0\ngini = 0.072\nsamples
= 1027\nvalue = [61, 1561, 0, 0]\nclass = 1'),
 Text(0.02287581699346405, 0.825, 'x[0] <= 3449.5\ngini = 0.027\nsamples =
631\nvalue = [14, 1000, 0, 0]\nclass = 1'),
 Text(0.0196078431372549, 0.775, 'gini = 0.0\nsamples = 212\nvalue = [0, 3
28, 0, 0]\nclass = 1'),
 Text(0.026143790849673203, 0.775, 'x[0] <= 3508.0\ngini = 0.04\nsamples =
419\nvalue = [14, 672, 0, 0]\nclass = 1'),
 Text(0.02287581699346405, 0.725, 'gini = 0.469\nsamples = 5\nvalue = [3,
5, 0, 0]\nclass = 1'),
 Text(0.029411764705882353, 0.725, 'x[0] <= 3812.0\ngini = 0.032\nsamples
= 414\nvalue = [11, 667, 0, 0]\nclass = 1'),
 Text(0.016339869281045753, 0.675, 'x[0] <= 3649.0\ngini = 0.07\nsamples =
120\nvalue = [7, 187, 0, 0]\nclass = 1'),
 Text(0.00980392156862745, 0.625, 'x[0] <= 3585.5\ngini = 0.028\nsamples =
```

In [170]:
```
rfc.fit(x_train,y_train)
```

```
C:\Users\P. VIJAY KUMAR\AppData\Local\Temp\ipykernel_4340\4070307935.py:1: Da
taConversionWarning: A column-vector y was passed when a 1d array was expecte
d. Please change the shape of y to (n_samples,), for example using ravel().
  rfc.fit(x_train,y_train)
```

Out[170]:
```
RandomForestClassifier()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [171]:
```
rfc.score(x_test,y_test)
```

Out[171]:
```
0.9341653666146645
```

CONCLUSION: Here i developed LinearRegression model, LogisticRegression model, Decision Tree model and RandomForest model for provided dataset.Among them the Decision Tree and RandomForest has got more accuracy on given dataset, So Decision Tree and RandomForest is best fittedmodel for our datset

In [ ]:
```
1
```